day04-网络请求

1. 网络请求

### axios请求

axios是一个开源免费的基于 Promise 的 HTTP 客户端，可以工作于浏览器中，也可以在 node.js 中使用。支持从浏览器中创建XML 请求，支持 node.js 中发出http请求，支持拦截请求和响应，支持转换请求和响应数据，支持自动转换JSON数据，客户端支持保护安全免受 CSRF/XSRF 攻击...

#### axios基本使用

安装 axios

npm install axios

在需要使用的文件中引入axios

import axios from 'axios'

常用方法

axios.request(config)

**axios.get(url[, config])**

axios.delete(url[, config])

axios.head(url[, config])

axios.options(url[, config])

**axios.post(url[, data[, config]])**

axios.put(url[, data[, config]])

axios.patch(url[, data[, config]])

#### axios配置,拦截器等应用

配置请求拦截器

axios.interceptors.request.use(function (config) {

// 可以给请求头添加token等字段, 完成用户验证

if (store.getters.Token) {

// 让每个请求携带token-- ['Token']为自定义key 请根据实际情况自行修改

config.headers['与后端约定的token访问key'] ='拼接需要的访问key' + store.getters.Token ;

}

return config;

}, function (error) {

// ...

return Promise.reject(error);

});

配置响应拦截器

axios.interceptors.response.use(function (response) {

// 所有 2xx 开头的响应状态码会触发该方法调用

// ...

return response;

}, function (error) {

// 所有非 2xx 开头的响应状态码会触发该方法调用

// ...

return Promise.reject(error);

});

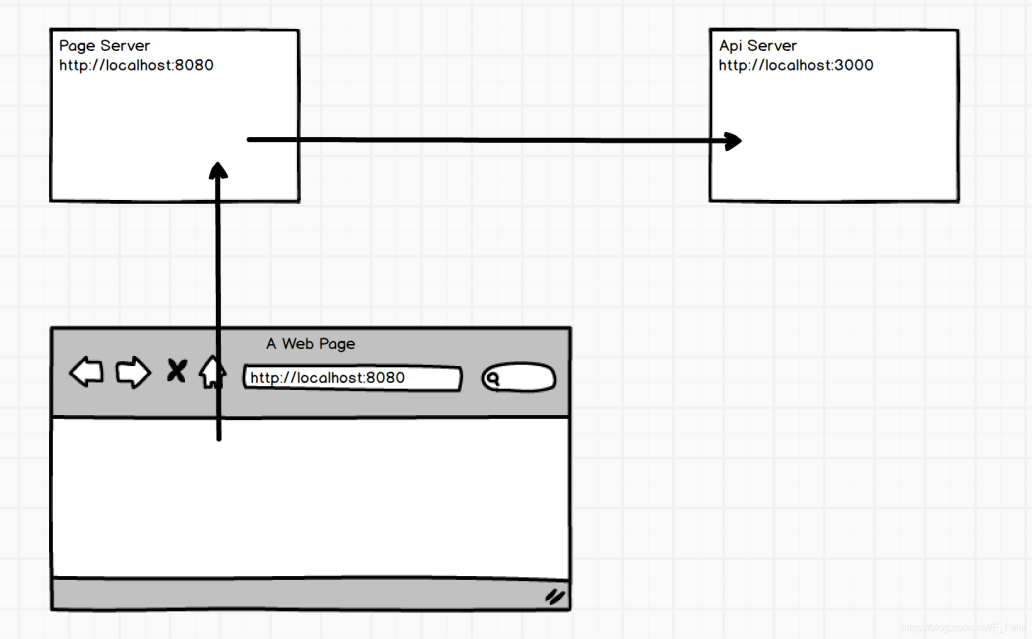
#### axios二次封装( api接口统一管理 )

### 反向代理（了解）

#### 反向代理用来解决什么问题?

我们通过 ajax 请求数据的时候存在同源策略问题，因为协议、域名、端口中有不同，就会导致产生跨域问题，而反向代理是我们开发中用的很多的一种解决跨域问题的方式。

跨域是浏览器的行为 当我们在浏览器端跨域请求数据的时候，其实浏览器已经把数据给我们返回了，但是浏览器自身检测到这是跨域的，因此把这部分数据拦截了，这就是所谓的跨域问题.



实际的原理是通过 服务器去请求数据，因为服务器之间是不存在跨域问题的，就如图二所示，页面是存在 8080 这台 page服务器上的 ，当我们请求数据的时候，让 8080 服务器帮我们做代理（即代理到目标服务器3000），让他去帮我们请求数据，这样就让 浏览器和请求地址就同源了，不会有跨域问题。

#### 反向代理的配置

在`src`目录下创建一个名称为`setupProxy.js`文件，为后续接口设置反向代理, 该配置会自动引入到webpack配置中.

const { createProxyMiddleware : proxy } = require("http-proxy-middleware");

module.exports = (app) => {

app.use(

"/api",

proxy({

// 此处的端口号要与后期数据请求的数据端一致

target: "http://localhost:9000",

changeOrigin: true,

})

);

};

1. 综合案例

<cdf-会员购> 分类页,全部商品页( 所需数据单独用node搭建服务端模拟, 配置代理解决跨域问题 )

1. 作业

<cdf-会员购> 所有商品页

1.用class组件 + jsx语法 封装组件;

2.使用单独的server服务器, 定义后端商品列表接口, 在合适的生命周期方法中请求数据;

3.对axios进行二次封装( 使用拦截器统一loading动画, 处理 token, 处理常见错误状态码 )

4.正确设置反向代理, 解决api接口跨域问题.

5. 100%还原页面样式.

6.封装列表组件List.

7.封装底部MyTabbar组件;

8.实现点击"销量","价格",...,按钮, 实现所有商品降序排序效果;

9.实现点击"销量","价格",...,按钮, 按钮高亮且只有一个高亮(红色);